Building Arabic Syntax Ontology based on Dependency Grammar Framework

Tarik elmalki

Abstract: Arabic Syntax Ontology is designed to represent, formally, most basic Grammatical Categories and their interrelationships that used in syntactic analyze of Arabic Grammar, Our ontological approach is based on two pillars ;first ASO linguistically is based on Dependency grammar framework analyzing sentence in term of set of labeled connection (Subject, object, …) on the other hand ASO is formally described in terms of Description Logics.

1.Introduction

Arabic Syntax Ontology (ASO for short)[1] is designed to represent, formally, most basic Grammatical Categories and their interrelationships that used in syntactic analyze of Arabic Grammar. More than just a controlled vocabulary of terms used in Arabic Grammar, ASO also attempts to provide tools for users to infer implicit facts from a set of asserted axioms. Thus, via inference system of ASO, one can deduce from subject relationships (العلاقة الفاعلية) between Verb and Noun that Nominative Case would be assigned to Noun, and the verb would be in the active form (مبني للمعلوم) , Furthermore, ASO gives a decision on whether its set of assertions is consistent, for example if we assign accusative case to name which occupies a subject position, then the reasoning system tell that such assertion is inconsistent with grammatical rules axiomatized in ASO.

ASO consists of a hierarchical description of main grammatical concepts (Noun, Verb, Particle, Tense, Phi-feature…), along with descriptions of their relationships (Subject, Object, Circumstance…), as well as constraints about these concepts and relationships (Fig **1**). So as to be parsed by a computer program, ASO is available in a machine-readable format.

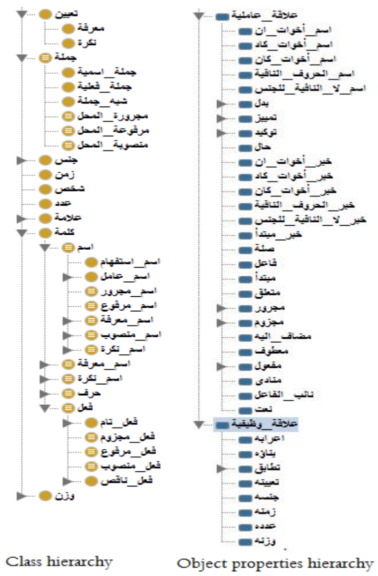

Fig 1: Grammatical Categories and syntactic relation

[1] -available at http://www.arabicontology.org/

ASO is based linguistically on Dependency grammar framework analyzing sentence in term of set of labeled connection (Subject, object, …) i.e: regent-dependents relationships holding between a words .

In this article we extend DG to deal with functional categories that force us to define another type of connection , not mentioned in dependency grammatical literature, it is functional relationships (HasTense, HasCase…).

After describing linguistically Traditional Arabic Grammars, we turn to formalize logically what we obtain by means of Description Logic constructors..

The remainder of this article is organized as follows. In Section 1 we introduce an overview of DL. Section 2 provides a formal description of Classical Arabic Grammar based on dependency grammar approach.

1-Description Logic Overview

We can identify two major objectives that DL fulfils. The first goal is to describe formally an application domain by specifying concepts (also known as classes), roles (also known as properties) and individuals (also known as objects) that are instances of these concepts. The second goal is provide a reasoning service which allows one to infer implicitly represented knowledge from the knowledge that is explicitly contained in the knowledge base.

So a knowledge base consists of two parts:

$$KB = \{ TBox , ABox \}$$

The terminological knowledge TBox (the vocabulary of an application domain) refers to classes of objects and their relationships, while the assertional knowledge ABox contains assertions about names individuals in terms of this vocabulary.

The vocabulary of TBox consists of concepts, viewed as unary predicate, denote sets of individuals, and roles, interpreted as binary predicate, and denote relationships between individuals.

1-1.Syntax

Concept descriptions in SHIQ are formed according to the following syntax rule:

if C , D are two concepts and r is a role and n is nonnegative integer, then, $C \sqcap D$, $C \sqcup D$, $\sim C$, $\forall r.C$, $\exists r.C$, $\geq nr.C$ *and* $\geq nr.C$ are also concepts *.*

The corresponding syntax for Web Ontology Language is shown in (**Fig 2**):

Abstract Syntax	DL Syntax	Semantics
Class(A)	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
Class(owl:Thing)	$\top$	$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
Class(owl:Nothing)	$\bot$	$\bot^{\mathcal{I}} = \emptyset$
intersectionOf($C_1, C_2, \ldots$)	$C_1 \sqcap C_2$	$(C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
unionOf($C_1, C_2, \ldots$)	$C_1 \sqcup C_2$	$(C_1 \sqcup C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
complementOf(C)	$\neg C$	$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
oneOf($o_1, o_2, \ldots$)	$\{o_1\} \sqcup \{o_2\}$	$(\{o_1\} \sqcup \{o_2\})^{\mathcal{I}} = \{o_1^{\mathcal{I}}, o_2^{\mathcal{I}}\}$
restriction(R someValuesFrom(C))	$\exists R.C$	$(\exists R.C)^{\mathcal{I}} = \{x \mid \exists y. \langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
restriction(R allValuesFrom(C))	$\forall R.C$	$(\forall R.C)^{\mathcal{I}} = \{x \mid \forall y. \langle x, y\rangle \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
restriction(R hasValue(o))	$\exists R.\{o\}$	$(\exists R.\{o\})^{\mathcal{I}} = \{x \mid \langle x, o^{\mathcal{I}}\rangle \in R^{\mathcal{I}}\}$
restriction(R minCardinality(m))	$\geqslant mR$	$(\geqslant mR)^{\mathcal{I}} = \{x \mid \sharp\{y. \langle x, y\rangle \in R^{\mathcal{I}}\} \geq m\}$
restriction(R maxCardinality(m))	$\leqslant mR$	$(\leqslant mR)^{\mathcal{I}} = \{x \mid \sharp\{y. \langle x, y\rangle \in R^{\mathcal{I}}\} \leq m\}$

Fig 2

1-2.Semantic

To provide a formal meaning for the above syntax , we need to interpret each concept, role and individual in term of model which consists of two parts: a non-empty set Δ^I , (the domain of the interpretation) and an interpretation function $.^I$, which assigns to every atomic concept A a set $A \in \Delta^I$ and to every atomic role R a binary relation Δ^I $\Delta^I \times$

$(C \sqcap D)^I$	$C^I \cap D^I$
$(C \sqcup D)^I$	$C^I \cup D^I$
$\sim C^I$	$\Delta^I \setminus C^I$
$(\exists r.C)^I$	$x \in \Delta^I / \exists y . (x,y) \in r^I \wedge y \in C$
$(\forall r.C)^I$	$x \in \Delta^I / \forall y . (x,y) \in r^I \rightarrow y \in C$

2- Traditional Arabic Grammars

Traditional Arabic Grammars (TAG) can be represented as a triple (*GC*, *R*, *AX*), where *GC* denotes Grammatical Categories, *R* denotes a set of Syntactic Relation between elements of *GC*, *AX* define axioms.

Hence, to every sentence of a language, is assigned two levels of syntactic description; the dependency level describes sentence's structure in term of directed arcs which are called dependencies or connection , each of these arcs links a dependent to a head or regent and each connexion is labeled with the role of the dependent in relation to the head.

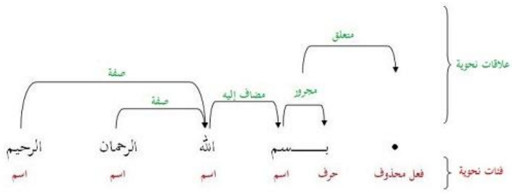

Fig 3:

At the other level of description, the categorical structure aims to provide a precise definition of each entity in term of **attribute–value matrix** which has two columns, one for the feature names (Number, Gender, Tense..) and the other for the values . For example the feature named Number might have either the value Singular (مفرد) or Plural(جمع) or Dual(مثنى), and the feature named Tense might have either Past(الماضي) or Present(الحاضر) or Future(المستقبل).

Each part of speech is specified with a determined matrix; the feature matrix for the Noun is written like this:

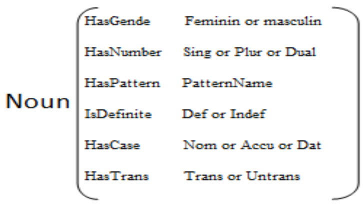

Fig 4

For the Verb:

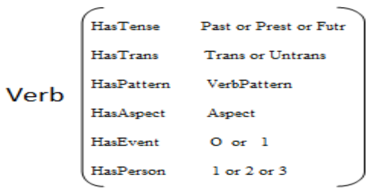

Fig 5

Feature Matrix of each lexical entity is inactive or unvalued, unless the entities merge with each other in the dependency level. Once the merging process is done, an appropriate value would be allocated to each feature name, otherwise the sentence will be ungrammatical.

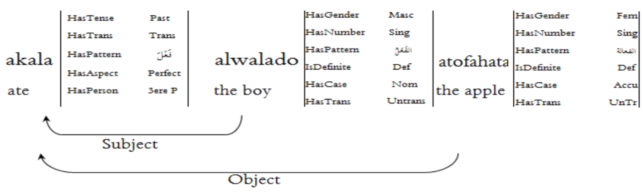

Fig 6

Both levels interact with each other, for example: syntactic dependencies govern Case and Voice feature:

Subject (Verb, Noun)→ HasCase(Noun,Nominative Case) ∧ HasVoice (Verb , Active)

At the other hand transitive feature governs Objective Dependency

HasTranse(Verb , Transitive)→ (Object (Verb, Noun) → HasCase(Noun, Accusative Case)).

Some feature govern other feature as in:

HasNumber(Noun , Dual) → HasPattern (Noun, DualPatternNoun)

2-1. Grammatical Categories:

Grammatical Categories (GC) can be divided into two subcategories: Lexical Categories (LC) and Functional Categories (FC):

- 1- GC = LC ∪ FC

The term 'Lexical Categories' is used here to cover what TAG call parts of speech, scholars of Arabic Grammar agree that speech is divided into three main categories, the main classes are Nouns (N) اسم, Verbs (V) فعل and Particles (P) حرف

- 2- LC = N ∪ V ∪ P

Functional Categories (FC) has no thematic content like Tense, Aspect, Template Pattern...and includes a collection of functional features into which lexical units can be broken down. Each feature is associated with a function relating lexical unit with appropriate feature-value.

- 3- FC = Tense ∪ Aspect ∪ Number ∪ Gender ∪ Pattern ∪ Transitive ∪ Voice ∪ Person ∪ Definiteness ∪ Case

- ✓ Tense

Tense is seen as a "*grammaticalisation of location in time* "[2], and can be realized linguistically in two different ways : Lexically by adding a prefix to a verb "س+يجلس" and grammatically by altering the morphological Pattern of the verb." جلس يجلس".

Tense feature includes tree basic values; Past, Present and Future

- 4- Tense = {Past , Present , Future}

As Tense can be realized particularly on verbs, we conceive a function HasTense that operates uniquely on verbs and associates appropriate Tense-value with each verb in the sentence.

- 5- HasTense (Verb , Tense)

Ex: HasTense (أكل , past)

HasTense (سيأكل , Future)

- ✓ Number

"'*Number' is a grammatical category which encodes quantification over entities or events denoted by nouns or nominal elements*"[3]

The Number feature applies to Nouns and their derivatives, and forms a closed set consisting of tree values:

- 6- Number = {Plural, Singular, Dual}

Number-values are assigned to Nouns via a function HasNumber():

- 7- HasNumber (Noun , Number)

Ex: HasNumber (رجل, Singular)

- ✓ Gender

Gender Feature pertains to nouns, morphologically, most Nouns that end with the morpheme 'a' are feminine, and there is two values for every noun :

- 8- Gender = {Masculine, Feminine}

Gender can be realized particularly on Nouns, thus we assume a function mapping each Noun to Gender-value:

- 9- HasGender (Noun, Gender)

Ex: HasGender (رجل, Masculine)

HasGender (امرأة, Feminine)

- ✓ Person

Feature Person refers to participants in an event and affects verbs, a value inventory for the feature person includes:

Person = { 1st person , 2nd person , 3rd person }

HasPerson (Verb , Person)

- ✓ Definiteness

In AG , Nouns may be either definite or indefinite :

- 10- Definiteness = {Definite, Indefinite}

[2]-http://www.grammaticalfeatures.net/features/tense.html

[3] - http://www.grammaticalfeatures.net/features/number.html

And realized via a function isDefinite():

- 11- IsDefinite (Noun, Definiteness)

Ex: IsDefinite (رجل, Indefinite)

Definiteness can be encoded using two ways; lexically, by adding a prefix ال to the beginning Noun as a prefix and syntactically via relating prefixless (without article ال) to definite nouns. So the prefixless noun receive his Definiteness through genitive relationship , in -12 - , although سيد is prefixless it is seen as definite noun because it occurs in genitive relation with a definite noun الدار

- 12 - دخل سيدُ الدارِ

✓ PATTERN

in Arabic language, Morphological derivation process is resulted from a combination of two abstract component: a lexical root (جذر) and a specific template pattern (وزن) , each pattern is associated with a previous meaning in such way that a word's meaning is as product of both level lexical and morphological pattern , thus the pattern فاعل denotes the one doing something , while مفعول denotes passive participle onto which the action is done.

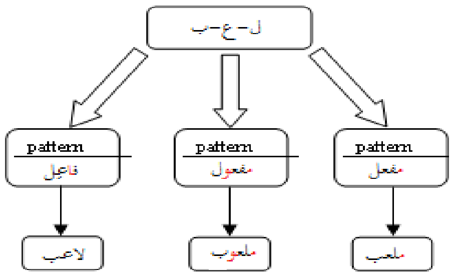

Fig 7

✓ TRANSITIVITY

Transitive feature is formal mechanism for representing object required by either particular verb or certain derivative Nouns. For example, the noun 'ضارب' (hitter) requires one object, whilst the verb 'أعطى' (give) requires two objects . This is encoded in the lexical entry for the verb. Certain verb has no object like جلس' (sit). So we have three values for Transitive feature:

- 13- Transitive = {0,1,2 }

So , HasTransitive() operates either on Noun or Verb by assigning to them Transitive-Value

- 14- HasTransitive (Noun,Transitive)

Example:

درهماً	زيداً	الرجلُ	أعطى
dirham.ACC	zaydan.ACC	Det.man.NOM	give.past
Object2	Object1	Subject	

HasTransitive(أعطى,2)

✓ CASE

Case can be defined as a system of marking nouns, verbs and clauses that reflects the syntactic function performed by those components in the sentence.

Encountered cases in TAG include four values:

- 15- Case = {Nominative, Accusative , Jussive , Dative}

- 16- HasCase(Noun,Case)

- 17- HasCase(Verb,Case)

Ex: HasCase(رجل, Nominative)

Ex : HasCase(رجل, Accusative)

2-2. Relationships

R in the triple above denotes a set of Syntactic Relation between elements of GC, Hence a sentence S can be defined formally in term of ordered pair of words:

- 18- $x,y \in LC \quad S = \wedge \; R(x,y)$

So , the sentence (- 19-) would be analyzed into two relational components as shown in (- **20**-)

- 19-

التفاحة	الولدُ	أكل
The apple-ACC	The boy-NOM	ate

The boy ate the apple

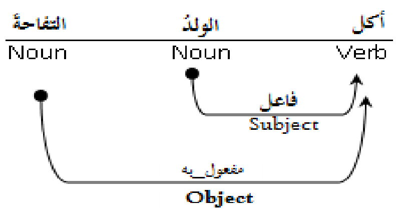

Fig 8

- 20- S= Subject (ate ,the boy) ∧ Object (ate ,the apple)

فاعل(أكل ،الولدُ) ∧ مفعول به (أكل،التفاحة)

These Relations are required to satisfy the following axioms:

1- Irreflexivity

Syntactic Relation is non-reflexive as no lexical element can stand in relation to itself:

- 21- $\forall \boldsymbol{x} \in \boldsymbol{LC} \quad \sim \boldsymbol{R}(\boldsymbol{x},\boldsymbol{x})$

~فاعل (أكل،أكل)

~Subject (ate , ate)

2- Asymmetric

Given a syntactic relation R , whenever it holds between two words x and y , never holds in the opposite direction:

- 22- $\forall \boldsymbol{x},\boldsymbol{y} \in \boldsymbol{LC} \quad \boldsymbol{R}(\boldsymbol{x},\boldsymbol{y}) \rightarrow \sim \boldsymbol{R}(\boldsymbol{y},\boldsymbol{x})$

فاعل (أكل،الولد) ← ~فاعل (الولد،أكل)

Subject (ate , the boy) → ~Subject (the boy , ate)

3- Non-transitivity

If we consider a syntactic relation R , whenever it holds between both x and y and between y and z , never holds between x and z .

- 23- $\forall \boldsymbol{x},\boldsymbol{y} \in \boldsymbol{LC} \quad \boldsymbol{R}(\boldsymbol{x},\boldsymbol{y}) \wedge \boldsymbol{R}(\boldsymbol{x},\boldsymbol{z}) \rightarrow \sim \boldsymbol{R}(\boldsymbol{x},\boldsymbol{z})$

4. There is unique governor x in the relation R which dominates y :

- 24- $\exists! x \in \boldsymbol{LC} \qquad R(x,y)$

1-3.Lexical and Function Relationships

Two types of relations must be distinguished according to roles that perform in the construction of sentence: Lexical Relation (LR) and Functional Relation (FR):

$$R = LR \cup FR$$

The former class of relation (LR) is charged with merging lexical categories:

- 25- $\forall (x, y) \in LC \quad LR(x,y)$

Whilst the latter class of relation (FR) maps a word to a set of feature-values (Tense, Case, Aspect, phi-feature..)

- 26- $(\forall x \in LC)\ (\exists y \in FC)\ FR(x,y)$

Thus nominative case would be assigned to word occurring in governed place:

- 27- **Subject (x,y)→HasCase (y, Nominative)**

Where x , y ∈ Lexical Categories, and Nominative is Case Feature

2-3.Entity Definition

One can summarize the inventory of feature that we need to describe formally TAG in the table below :

Feature	Value	
Gender	Masculine Feminine	HasGender(Noun,Gender)
Person		HasPerson(Verb,Person)
Number	Plural, Singular Dual	HasNumber(Noun,Number)
Definiteness	Definite Indefinite	IsDefinite(Noun, Definiteness)
Case	Nominative Accusative Jussive Dative	HasCase(Noun,Case) HasCase(Verb,Case)
Tense	Past Present Future	HasTense(Verb, Tense)
Transitive	transitive,intransitive	HasTransitive(Verb,Transitive)
Pattern	VerbPattern NounPattern	HasPattern(Verb,VerbPatter) HasPattern(Verb,NounPatter)

- Noun definition:

Using constructors of Description Logics we can describe a Noun as :

Noun ≡ ∀ HasGender.Gender ⊓ ∀ HasNumber.Number ⊓ ~ ∀ HasTense.Tense ⊓ ∀**HasCase.Case** ⊓ ∃HasTransitive.Transitive ⊓ ∀IsDefinite. Definiteness .

We can define Nominative noun and accusative noun in the same way:

NominativeNoun ≡ ∀**HasCase.** Nominative ⊓ Noun

AccusativeNoun ≡ ∀**HasCase.** Accusative ⊓ Noun

- Verb definition:

Verb ≡ ~∀ HasGender.Gender ⊓~ ∀ HasNumber.Number ⊓ ∀ HasTense.Tense ⊓ ∃**HasCase.Case** ⊓ ∀HasTransitive.Transitive ⊓ ∀ **HasVoice.Voice** ⊓ ∀ **HasMood.Mood**

- Particle Definition

Particle ≡ ~∀ HasGender.Gender ⊓~ ∀ HasNumber.Number ⊓~ ∀ HasTense.Tense ⊓~ ∀ **HasCase.Case**

We are in position to define certain syntactic relationships at the dependency level , hence It follows from the above that certain Subject relationship can be

redefined as syntactic relation whose domain is verbs or Active Participle and range is NominativeNoun

- 28- Subject (verb , NominativeNoun)

In the same way we can define syntactic object :

Object (verb , AccusativeNoun)

3-Using SPARQL to Query Arabic Syntax Ontology

SPARQL is a SQL-like language for querying ASO data which can be considered in SQL relational database terms as a table with three columns – the subject column, the predicate column, and the object column, for querying this table we use expressions with variables:

Suppose we want to retrieve all possible relationships R that hold between nominative noun and governor X , this query can be captured in terms of Description Logics as (- **29**-):

- 29- R(X, NominativeNoun)

The result query is shown in (Fig 9)

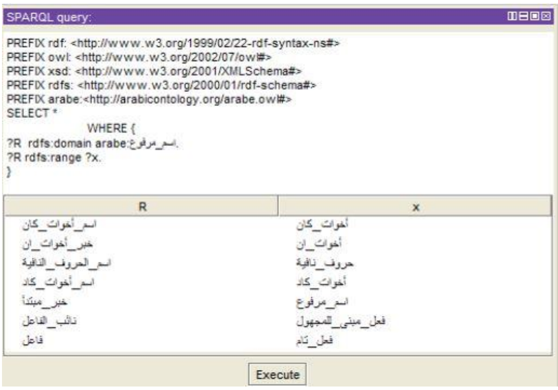

Fig 9

If we look for a specific governor for example particle governor we can write a query like ()

- 30- R(X, NominativeNoun) ∧ Particle(X)

We obtain the result (**Fig 10**) :

Fig 10

References

[1] Amedeo Napoli, Une introduction aux logiques de description , INRIA , 1997.

[2] D. G. HAYS, Grouping and dependency theories. P-1910, RAND Corporation, 1960

[3] Franz Baader , Diego Calvanese , Deborah L. McGuinness , Daniele Nardi , Peter F. Patel-Schneider THE DESCRIPTION LOGIC HANDBOOK: Theory, implementation, and applications. Cambridge University Press; 2 edition (June 28, 2010).

[4] Franz Baader, Ian Horrocks, Ulrike Sattler. Description Logics as Ontology Languages for the Semantic Web, Mechanizing Mathematical Reasoning, Volume 2605 of the series Lecture Notes in Computer Science pp 228-248.

[5] J.Pan , Description Logics : Reasonbing Support For The Semantic Web.Phd Thesis 2004.

[6] Jonathan Owens (1984). Structure, Class and Dependency: Modern Linguistic Theory and the Arabic Grammatical Tradition. Lingua, 64:1 (25-62).

[7] Jonathan Owens (1988). The Foundations of Grammar: An Introduction to Medieval Arabic Grammatical Theory. John Benjamins.

[8] Jonathan Owens (1989). The Syntactic Basis of Arabic Word Classification. Arabica, 36:2 (211-234).

[9] Kais Dukes , Statistical Parsing by Machine Learning from a Classical Arabic Treebank, The University of Leeds School of Computing, September, 2013- in http://www.kaisdukes.com/papers/thesis-dukes2013.pdf

[10] Lucien Tesnière ,Eléments de syntaxe structurale, Préface de Jean Fourquet , Klincksieck,1959.

[11] Melcuk, Igor A. Dependency Syntax : Theory and Practice . State University of New York Press. 1988.

[12] Mustafa Jarrar: The Arabic Ontology Applied Ontology Journal. IOS Press .

[13] Robinson, J. J. (1970), 'Dependency structures and transformation rules', Language , 46, 259–285.

[14] Sylvain Kahane , Grammaire d'unification sens-texte: Vers un modèle mathématique articulé de la langue naturelle Document de synthèse de l'habilitation à diriger les recherches,Université Denis Diderot, Paris VII.in https://hal.archives-ouvertes.fr/hal-00144032/document.

[15] Sylvain Kahane, Grammaires de dépendance formelles et théorie Sens-Texte,TALN 2001 ,(version revue et corrigée, juillet 2011). In http://www.atala.org/taln_archives/TALN/TALN-2001/taln-2001-tutoriel-003.pdf

[17] vilmos Ágel and klaus fischer. Dependency grammar and valency theory. the oxford handbook of LINGUISTIC ANALYSIS second edition.p225-258